Quantum Gates - Hardware Implementation

ECE 572, Fall 2001
Goran Negovetic
Quantum Gates

- Quantum Circuits (QC) are composed of quantum gates
- Quantum computation results in exponential amount of calculation in a polynomial amount of space and time
- This leads to Quantum Parallelism
- How to implement QCs (finite precision, observation etc).
Qubit

- Qubit is the elementary unit of storage
- Qubit has three states: zero, one, and superposition of zero and one
- Superposition is expressed as amplitude of complex probability
- $2^N$ superposition states for $N$ Qubits
Quantum Gates

- Quantum gates can be represented as matrix transformations.
- Multiplication of $2^N$ vector by $2^N \times 2^N$ matrix.
- One bit unitary operations:

$$
X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad Y = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \quad W = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}
$$
Quantum Gates (cont)

- Many quantum operations can be implemented as register content transfer
- Controlled NOT (XOR) gate:

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} \times \begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
a_3 \\
\end{bmatrix} = \begin{bmatrix}
a_0 \\
a_1 \\
a_3 \\
a_2 \\
\end{bmatrix}
\]

Swap Amplitudes \(a_2\) and \(a_3\)
Matrix Operations

- Quantum operations consist of:
  1. Complex number multiplication
  2. Kronecker product
  3. Register content transfer
  4. Matrix splitting
Matrix Operations (cont)

- Each elementary gate is a 2x2 or 4x4 matrix.
- Operation on a large vector space is broken into smaller chunks:
Example 1

\[
\text{Had} = \sqrt{2} * \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}; \\
\text{inv} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \quad \text{%Pauli_X} \\
\text{Pauli}_Y = \begin{bmatrix} 0 & -1i \\ 1i & 0 \end{bmatrix}; \\
\text{Pauli}_Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix};
\]

\[
\text{phase} = \begin{bmatrix} 1 & 0 \\ 0 & 1i \end{bmatrix}; \\
\text{pi}_8 = \begin{bmatrix} 1 & 0 \\ 0 & \exp((\pi/4)*i) \end{bmatrix};
\]

\[
% 2 \times 2 \\
\text{xor} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}; \\
\text{swap} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix};
\]

\[
\text{conZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}; \\
\text{conPhase} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix};
\]
Example 2
 Proposed design flow

- Quantum Logic Function
- Minimization (Synthesis)
- Quantum Circuit Net-list
- Logic Gate Instantiation
- Binary Logic Synthesis & Implementation
Design Flow (cont)

- My goal:
  1. Implement quantum gate primitives in Verilog
  2. Implement other necessary matrix operations (Kronicker product, complex multiplication etc.)
  3. Make a utility program to translate the Quantum circuit net-list to the library gate instantiation.
Design considerations

- Data representation
  - sign 15-bit fraction
- Unsigned multiplication and sign logic
- 1, 0.99998, … , 0.0000305, 0
- Virtex II architecture: up to 8M gates, 1.5 Mb SRAM, 18 x 18-bit multipliers, CLA adders
Conclusion

- Limitation: Simulation of quantum circuits is memory/resource exhaustive
- Similarity to unitary transforms used for Signal Processing
Acknowledgment

1. A Parallel Quantum Computer Simulator, Kevin M. Obanland and Alvin M. Despain
2. Quantum Computation: Theory and Implementation, Edward S. Boyden
3. Efficient Quantum Transforms, Peter Hoyer
4. Elementary Gate for Quantum Computation, Adriano Barenco et al.